promise 44 no longer conforms to the business constraints specified in component ATP request 32, LFM 22 may generate an annotated or other appropriate failure notification for communication to fulfillment server 16. Example annotations may include "insufficient product quantities" or "unable to meet shipment delivery or lot sizing requirements."

In one embodiment, component promise 44 is an object having the following attributes or otherwise supporting the following information, in any suitable combination and without limitation: (1) *component promise ID* - assigned when LFM 22 and/or ATP server 14 creates the component promise and may be identical to the *component quotation ID*; (2) *fulfillment server ID*; (3) *accept-by* - may indicate an expiration date for component promise by which an acceptance must be received or corresponding promise reservations may be released; (4) *component promise status* - indicates whether the component promise has succeeded or failed; and (5) *failure reason* - brief description of reason for the promise having failed, if any.

Process Component Promises [Fulfillment Server]

Once fulfillment server 16 has processed component quotation confirmations 42 and sent them to LFMs 22, it monitors completion of the resulting component promises 44. In one embodiment, promise 46 is considered complete when each of the originating component quotation confirmations 42 has received one or more component promises 44 or failure notifications. Fulfillment server 16 may also monitor the *promise by* attribute specifying the length of time fulfillment server 16 should wait for component promises 44 from LFMs 22. When this constraint is reached before all the component promises 44 have been received, such that the quotation confirmation 40 has essentially expired, fulfillment server 16 may generate an appropriate failure notification and send it to client 12. In formulating promise 46, fulfillment server 16 may take into account any *accept-by* attributes for component promises 44 and specify an *accept-by* attribute for promise 46 accordingly.

In one embodiment, once a component promise 44 expires, fulfillment server 16 and appropriate LFMs 22 release reservations of supply. Where fulfillment server 16 provided a stricter *accept_by* date than LFMs 22, fulfillment server 16 may need to send an indication of the release back to LFMs 22. Similarly, if promise 46 fails or

gets rejected, fulfillment server 16 notifies LFMs 22 so that LFMs 22 can release suitable supply reservations.

When fulfillment server 16 receives component promises 44 from LFMs 22 and promise 44 is complete, fulfillment server 16 evaluates the overall promise 44 according to the business constraints specified in the original ATP request 30 to again evaluate the success of the overall response. This is done again during the promise generation phase because it is possible that one or more component promises 44 may be different from the original component quotations 34. Pricing may also need to be calculated again during the promise generation phase if any component promises 44 are different than original component quotations 34. In addition, if there were multiple component quotations 34 for a particular component ATP request 32, it may be necessary to calculate a final confirmed price. In one embodiment, all of the component promises 44 must be valid to achieve a successful promise 46.

In one embodiment, fulfillment server 16 may mutate or otherwise manipulate component promises 44, according to the information and rules they provide, such that together component promises 44 satisfy the business constraints applied at fulfillment server 16 or asserted along with ATP request 30. In addition to sending promise 44 to client 12, fulfillment server 16 may send the mutated component promises 44 back to originating LFMs 22, such that the LFMs 22 may adjust their reservations of supply appropriately.

If the overall response no longer meets requirements of ATP request 30, due to changes in product availability in the interval between quotation 36 and promise 46 or for any other reason, fulfillment server 16 may assign a failure status to promise 46 and annotate it with descriptive information before sending the promise 46 to client 12. Fulfillment server 16 may simply pass along failure status annotations received from LFMs 22. Instead or in addition, fulfillment server 16 may assign an annotation of its own. For example, although LFM 22 may have generated an acceptable component promise 44, fulfillment server 16 may determine that promise 46 fails overall based on promise pricing not meeting specified business constraints for the customer.

Fulfillment server 16 may include a delivery coordination engine component, depending on requirements of the customer. Without this capability, fulfillment

server 16 would return the optimal shipment dates from the respective manufacturing and distribution locations rather than returning the delivery date to the customer. Delivery coordination may be accomplished using a relatively simple table-driven technique that links products, locations, and standard lead times. More sophisticated implementations may involve the use of an advanced planning engines for transportation and logistics optimization. In this scenario, it is likely that delivery coordination may be calculated in multiple phases. For example, a table-based standard lead time approach may be used in the initial promise generation phase to derive a preliminary delivery date. Because transportation planning optimization is generally most effective when the requirements of multiple deliveries are considered, a second phase of batch-oriented processing may be desirable to evaluate the entire request backlog. As a result of such second-phase processing, the delivery dates corresponding to the individual ATP requests 30 may be adjusted to reflect an optimized overall delivery plan.

In another embodiment, fulfillment server 16, LFM 22, and/or other components of system 10 may communicate with a delivery engine to support transport, delivery, and tracking of product shipments. For example, fulfillment server 16 could communicate with a TRADE MATRIX GLOBAL LOGISTICS MONITOR from i2 TECHNOLOGIES, INC. In yet another embodiment, fulfillment server 16 may communicate with a delivery system used by a carrier that provides logistics services. For example, fulfillment server 16 could query the carriers' systems to identify the cost and availability of various delivery services. Using this information, fulfillment server 16 could select a service and arrange for shipment. If needed, fulfillment server 16 could information the carrier's system of the needed pick-up and/or delivery date.

When fulfillment server 16 has completed evaluating promise 46, has calculated pricing and delivery as appropriate, and the overall response is still deemed satisfactory, then fulfillment server 16 sends promise 46 (including all valid component promises 44) to client 12 for confirmation. The structure of promise 46 models that of the originating quotation 36, but is potentially simpler than its quotation counterpart since quotation 36 may have been multi-dimensional whereas